Resum

Aquest document inclou quatre annexes que formen part del projecte. El primer d'ells és el conjunt de jocs de dades utilitzats durant l'experiència computacional, mentre que el segon annex són els resultats d'aquesta experiència.

El tercer annex són les dades utilitzades per al càlcul de l'impacte ambiental del projecte i finalment l'últim annex inclou els algorismes utilitzats en el projecte.

Sumari

Annex A. Jocs de dades estudiats.

La següent enumeració de jocs de dades utilitzats per a trobar els moviments òptims del robot segueixen la següent estructura:

nmaquines
TA[ ]
TB[ ]
tple
tbuit

5
25 34 79 48 25
75 30 34 26 48
12
8

5
25 0 79 48 25
75 30 34 26 48
12
8

5
25 34 79 48 25
75 30 0 26 48
12
8

5
0 34 79 48 25
75 30 34 26 48
12
8

5
25 34 79 48 25
75 0 34 26 48
12
8

5
25 34 79 0 25
75 30 34 26 48
12
8

5
25 34 79 48 25
0 30 34 26 48
12
8

5
25 34 0 48 25
75 30 34 26 48
12
8

5
25 34 79 48 25
75 30 34 0 48
12
8

```
5
25 34 79 48  0
75 30 34 26 48
12
8
```

```
5
46 00 79 24 35
43 22 40 49 64
10
7
```

```
5
46 51 79 24 35
43 22 40 00 64
10
7
```

```
5
25 34 79 48 25
75 30 34 26  0
12
8
```

```
5
46 51 79 24 35
43 00 40 49 64
10
7
```

```
5
46 51 79 24 00
43 22 40 49 64
10
7
```

```
5
46 51 79 24 35
43 22 40 49 64
10
7
```

```
5
46 51 00 24 35
43 22 40 49 64
10
7
```

```
5
46 51 79 24 35
43 22 40 49 00
10
7
```

```
5
00  51 79 24 35
43 22 40 49 64
10
7
```

```
5
46 51 79 24 35
43 22 00 49 64
10
7
```

```
5
50 57 51 44 54
72 27 79 40 46
13
9
```

```
5
46 51 79 24 35
00 22 40 49 64
10
7
```

```
5
46 51 79 00 35
43 22 40 49 64
10
7
```

```
5
00  57 51 44 54
72 27 79 40 46
13
9
```

5
50 57 51 44 54
00 27 79 40 46
13
9

5
50 57 51 00 54
72 27 79 40 46
13
9

5
00 80 60 80 28
49 71 54 53 56
12
8

5
50 00 51 44 54
72 27 79 40 46
13
9

5
50 57 51 44 54
72 27 79 00 46
13
9

5
50 80 60 80 28
00 71 54 53 56
12
8

5
50 57 51 44 54
72 00 79 40 46
13
9

5
50 57 51 44 00
72 27 79 40 46
13
9

5
50 00 60 80 28
49 71 54 53 56
12
8

5
50 57 00 44 54
72 27 79 40 46
13
9

5
50 57 51 44 54
72 27 79 40 00
13
9

5
50 80 60 80 28
49 00 54 53 56
12
8

5
50 57 51 44 54
72 27 00 40 46
13
9

5
50 80 60 80 28
49 71 54 53 56
12
8

5
50 80 00 80 28
49 71 54 53 56
12
8

```
5
50 80 60 80 28
49 71 00 53 56
12
8
```

```
5
55 57 43 67 62
43 22 40 49 64
10
7
```

```
5
55 57 00 67 62
43 22 40 49 64
10
7
```

```
5
50 80 60 00 28
49 71 54 53 56
12
8
```

```
5
00 57 43 67 62
43 22 40 49 64
10
7
```

```
5
55 57 43 67 62
43 22 00 49 64
10
7
```

```
5
50 80 60 80 28
49 71 54 00 56
12
8
```

```
5
55 57 43 67 62
00 22 40 49 64
10
7
```

```
5
55 57 43 00 62
43 22 40 49 64
10
7
```

```
5
50 80 60 80 00
49 71 54 53 56
12
8
```

```
5
55 00 43 67 62
43 22 40 49 64
10
7
```

```
5
55 57 43 67 62
43 22 40 00 64
10
7
```

```
5
50 80 60 80 28
49 71 54 53 00
12
8
```

```
5
55 57 43 67 62
43 00 40 49 64
10
7
```

```
5
55 57 43 67 00
43 22 40 49 64
10
7
```

```
5
55 57 43 67 62
43 22 40 49 00
10
7
```

```
5
58 59 78 76 49
43 00 40 49 64
10
7
```

```
5
58 59 78 76 00
43 22 40 49 64
10
7
```

```
5
58 59 78 76 49
43 22 40 49 64
10
7
```

```
5
58 59 00 76 49
43 22 40 49 64
10
7
```

```
5
58 59 78 76 49
43 22 40 49 00
10
7
```

```
5
00 59 78 76 49
43 22 40 49 64
10
7
```

```
5
58 59 78 76 49
43 22 00 49 64
10
7
```

```
5
77 57 67 45 27
49 71 54 53 56
12
8
```

```
5
58 59 78 76 49
00 22 40 49 64
10
7
```

```
5
58 59 78 00 49
43 22 40 49 64
10
7
```

```
5
00 57 67 45 27
49 71 54 53 56
12
8
```

```
5
58 00 78 76 49
43 22 40 49 64
10
7
```

```
5
58 59 78 76 49
43 22 40 00 64
10
7
```

```
5
77 57 67 45 27
00 71 54 53 56
12
8
```

5
77 00 67 45 27
49 71 54 53 56
12
8

5
77 57 67 45 27
49 71 54 00 56
12
8

5
20 49 22 68 67
00 67 37 71 42
15
10

5
77 57 67 45 27
49 00 54 53 56
12
8

5
77 57 67 45 00
49 71 54 53 56
12
8

5
20 00 22 68 67
71 67 37 71 42
15
10

5
77 57 00 45 27
49 71 54 53 56
12
8

5
77 57 67 45 27
49 71 54 53 00
12
8

5
20 49 22 68 67
71 00 37 71 42
15
10

5
77 57 67 45 27
49 71 00 53 56
12
8

5
20 49 22 68 67
71 67 37 71 42
15
10

5
20 49 00 68 67
71 67 37 71 42
15
10

5
77 57 67 00 27
49 71 54 53 56
12
8

5
00 49 22 68 67
71 67 37 71 42
15
10

5
20 49 22 68 67
71 67 00 71 42
15
10

```
5
20 49 22 00 67
71 67 37 71 42
15
10
```

```
5
00 41 74 35 23
71 67 37 71 42
15
10
```

```
5
66 41 74 35 23
71 67 00 71 42
15
10
9
```

```
5
20 49 22 68 67
71 67 37 00 42
15
10
```

```
5
66 41 74 35 23
00 67 37 71 42
15
10
```

```
5
66 41 74 00 23
71 67 37 71 42
15
10
```

```
5
20 49 22 68 00
71 67 37 71 42
15
10
```

```
5
66 00 74 35 23
71 67 37 71 42
15
10
```

```
5
66 41 74 35 23
71 67 37 00 42
15
10
9
```

```
5
20 49 22 68 67
71 67 37 71 00
15
10
```

```
5
66 41 74 35 23
71 00 37 71 42
15
10
```

```
5
66 41 74 35 00
71 67 37 71 42
15
10
```

```
5
66 41 74 35 23
71 67 37 71 42
15
10
```

```
5
66 41 00 35 23
71 67 37 71 42
15
10
```

```
5
66 41 74 35 23
71 67 37 71 00
15
10
```

6
60 63 80 71 62 72
38 63 45 34 37 78
12
8

6
60 63 00 71 62 72
38 63 45 34 37 78
12
8

6
60 63 80 71 62 72
38 63 45 34 00 78
12
8

6
00 63 80 71 62 72
38 63 45 34 37 78
12
8

6
60 63 80 71 62 72
38 63 00 34 37 78
12
8

6
60 63 80 71 62 00
38 63 45 34 37 78
12
8

6
60 63 80 71 62 72
00 63 45 34 37 78
12
8

6
60 63 80 00 62 72
38 63 45 34 37 78
12
8

6
60 63 80 71 62 72
38 63 45 34 37 00
12
8

6
60 00 80 71 62 72
38 63 45 34 37 78
12
8

6
60 63 80 71 62 72
38 63 45 00 37 78
12
8

6
30 48 70 45 24 72
73 38 69 23 64 49
9
6

6
60 63 80 71 62 72
38 00 45 34 37 78
12
8

6
60 63 80 71 00 72
38 63 45 34 37 78
12
8

6
00 48 70 45 24 72
73 38 69 23 64 49
9
6

6
30 48 70 45 24 72
00 38 69 23 64 49
9
6

6
30 48 70 00 24 72
73 38 69 23 64 49
9
6

6
30 48 70 45 24 72
73 38 69 23 64 00
9
6

6
30 00 70 45 24 72
73 38 69 23 64 49
9
6

6
30 48 70 45 24 72
73 38 69 00 64 49
9
6

6
30 59 44 45 24 32
73 38 69 23 64 49
9
6

6
30 48 70 45 24 72
73 00 69 23 64 49
9
6

6
30 48 70 45 00 72
73 38 69 23 64 49
9
6

6
00 59 44 45 24 32
73 38 69 23 64 49
9
6

6
30 48 00 45 24 72
73 38 69 23 64 49
9
6

6
30 48 70 45 24 72
73 38 69 23 00 49
9
6

6
30 59 44 45 24 32
00 38 69 23 64 49
9
6

6
30 48 70 45 24 72
73 38 00 23 64 49
9
6

6
30 48 70 45 24 00
73 38 69 23 64 49
9
6

6
30 00 44 45 24 32
73 38 69 23 64 49
9
6

6
30 59 44 45 24 32
73 00 69 23 64 49
9
6

6
30 59 44 45 00 32
73 38 69 23 64 49
9
6

6
00 79 36 25 55 63
73 55 69 44 34 48
9
6

6
30 59 00 45 24 32
73 38 69 23 64 49
9
6

6
30 59 44 45 24 32
73 38 69 23 00 49
9
6

6
22 79 36 25 55 63
00 55 69 44 34 48
9
6

6
30 59 44 45 24 32
73 38 00 23 64 49
9
6

6
30 59 44 45 24 00
73 38 69 23 64 49
9
6

6
22 00 36 25 55 63
73 55 69 44 34 48
9
6

6
30 59 44 00 24 32
73 38 69 23 64 49
9
6

6
30 59 44 45 24 32
73 38 69 23 64 00
9
6

6
22 79 36 25 55 63
73 00 69 44 34 48
9
6

6
30 59 44 45 24 32
73 38 69 00 64 49
9
6

6
22 79 36 25 55 63
73 55 69 44 34 48
9
6

6
22 79 00 25 55 63
73 55 69 44 34 48
9
6

6
22 79 36 25 55 63
73 55 00 44 34 48
9
6

6
22 79 36 25 55 00
73 55 69 44 34 48
9
6

6
31 00 56 71 49 72
73 38 69 23 64 49
9
6

6
22 79 36 00 55 63
73 55 69 44 34 48
9
6

6
22 79 36 25 55 63
73 55 69 44 34 00
9
6

6
31 29 56 71 49 72
73 00 69 23 64 49
9
6

6
22 79 36 25 55 63
73 55 69 00 34 48
9
6

6
31 29 56 71 49 72
73 38 69 23 64 49
9
6

6
31 29 00 71 49 72
73 38 69 23 64 49
9
6

6
22 79 36 25 00 63
73 55 69 44 34 48
9
6

6
00 29 56 71 49 72
73 38 69 23 64 49
9
6

6
31 29 56 71 49 72
73 38 00 23 64 49
9
6

6
22 79 36 25 55 63
73 55 69 44 00 48
9
6

6
31 29 56 71 49 72
00 38 69 23 64 49
9
6

6
31 29 56 00 49 72
73 38 69 23 64 49
9
6

6
31 29 56 71 49 72
73 38 69 00 64 49
9
6

6
31 29 56 71 00 72
73 38 69 23 64 49
9
6

6
31 29 56 71 49 72
73 38 69 23 00 49
9
6

6
31 29 56 71 49 00
73 38 69 23 64 49
9
6

6
31 29 56 71 49 72
73 38 69 23 64 00
9
6

5
28 40 26 29 43
28 61 26 32 57
20
10

5
0 40 26 29 43
28 61 26 32 57
20
10

5
28 40 26 0 43
0 61 26 32 57
20
10

5
28 0 26 29 43
28 61 26 32 57
20
10

5
28 40 26 29 43
28 0 26 32 57
20
10

5
28 40 0 29 43
28 61 26 32 57
20
10

5
28 40 26 29 43
28 61 0 32 57
20
10

5
28 40 26 0 43
28 61 26 32 57
20
10

5
28 40 26 29 43
28 61 26 0 57
20
10

5
28 40 26 29 0
28 61 26 32 57
20
10

```
5
28 40 26 29 43
28 61 26 32 0
20
10
```

```
6
39 65 78 69 24 35
49 0 56 37 66 75
18
9
```

```
6
39 65 78 69 0 35
49 56 56 37 66 75
18
9
```

```
6
39 65 78 69 24 35
49 56 56 37 66 75
18
9
```

```
6
39 65 0 69 24 35
49 56 56 37 66 75
18
9
```

```
6
39 65 78 69 24 35
49 56 56 37 0 75
18
9
```

```
6
0 65 78 69 24 35
49 56 56 37 66 75
18
9
```

```
6
39 65 78 69 24 35
49 56 0 37 66 75
18
9
```

```
6
39 65 78 69 24 0
49 56 56 37 66 75
18
9
```

```
6
39 65 78 69 24 35
0 56 56 37 66 75
18
9
```

```
6
39 65 78 0 24 35
49 56 56 37 66 75
18
9
```

```
6
39 65 78 69 24 35
49 56 56 37 66 0
18
9
```

```
6
39 0 78 69 24 35
49 56 56 37 66 75
18
9
```

```
6
39 65 78 69 24 35
49 56 56 0 66 75
18
9
```

```
6
52 29 33 46 67 71
41 75 33 47 77 68
16
8
```

6
0 29 33 46 67 71
41 75 33 47 77 68
16
8

6
52 29 33 46 67 71
41 75 0 47 77 68
16
8

6
52 29 33 46 67 0
41 75 33 47 77 68
16
8

6
52 29 33 46 67 71
0 75 33 47 77 68
16
8

6
52 29 33 0 67 71
41 75 33 47 77 68
16
8

6
52 29 33 46 67 71
41 75 33 47 77 0
16
8

6
52 0 33 46 67 71
41 75 33 47 77 68
16
8

6
52 29 33 46 67 71
41 75 33 0 77 68
16
8

6
33 32 44 23 53 56
55 28 62 39 60 56
14
7

6
52 29 33 46 67 71
41 0 33 47 77 68
16
8

6
52 29 33 46 0 71
41 75 33 47 77 68
16
8

6
0 32 44 23 53 56
55 28 62 39 60 56
14
7

6
52 29 0 46 67 71
41 75 33 47 77 68
16
8

6
52 29 33 46 67 71
41 75 33 47 0 68
16
8

6
33 32 44 23 53 56
0 28 62 39 60 56
14
7

6
33 0 44 23 53 56
55 28 62 39 60 56
14
7

6
33 32 44 23 53 56
55 28 62 0 60 56
14
7

6
33 32 44 23 53 56
55 0 62 39 60 56
14
7

6
33 32 44 23 0 56
55 28 62 39 60 56
14
7

6
33 32 0 23 53 56
55 28 62 39 60 56
14
7

6
33 32 44 23 53 56
55 28 62 39 0 56
14
7

6
33 32 44 23 53 56
55 28 0 39 60 56
14
7

6
33 32 44 23 53 0
55 28 62 39 60 56
14
7

6
33 32 44 0 53 56
55 28 62 39 60 56
14
7

6
33 32 44 23 53 56
55 28 62 39 60 0
14
7

Annex B. Resultats de les simulacions.

Disposició Lineal

LINEAL					
Simulació	Màquina	Temps cicle original	Temps màquina	g.absolut	g.relatiu
1	1	328	25	24	0,96
2	1	285	46	26	0,565217391
3	1	368	50	20	0,4
4	1	332	50	36	0,72
5	1	276	55	40	0,727272727
6	1	285	58	31	0,534482759
7	1	325	77	43	0,558441558
8	1	395	20	49	2,45
9	1	395	66	45	0,681818182
10	1	389	60	36	0,6
11	1	291	30	27	0,9
12	1	291	30	27	0,9
13	1	304	22	14	0,636363636
14	1	303	31	17	0,548387097
15	1	430	28	56	2
16	1	477	39	56	1,435897436
17	1	424	52	41	0,788461538
18	1	380	33	26	0,787878788
1	1	328	75	20	0,266666667
2	1	285	43	17	0,395348837
3	1	368	72	20	0,277777778
4	1	332	49	20	0,408163265
5	1	276	43	19	0,441860465
6	1	285	43	17	0,395348837
7	1	325	49	20	0,408163265
8	1	395	71	19	0,267605634
9	1	395	71	25	0,352112676
10	1	389	38	28	0,736842105
11	1	291	73	15	0,205479452
12	1	291	73	15	0,205479452
13	1	304	73	14	0,191780822
14	1	303	73	17	0,232876712
15	1	430	28	36	1,285714286
16	1	477	49	29	0,591836735
17	1	424	41	25	0,609756098
18	1	380	55	14	0,254545455
1	2	328	34	28	0,823529412
2	2	285	51	19	0,37254902
3	2	368	57	11	0,192982456
4	2	332	80	12	0,15

5	2	276	57	24	0,421052632
6	2	285	59	24	0,406779661
7	2	325	57	12	0,210526316
8	2	395	49	-1	-0,020408163
9	2	395	41	15	0,365853659
10	2	389	63	28	0,444444444
11	2	291	48	10	0,208333333
12	2	291	59	10	0,169491525
13	2	304	79	7	0,088607595
14	2	303	29	12	0,413793103
15	2	430	40	28	0,7
16	2	477	65	34	0,523076923
17	2	424	29	17	0,586206897
18	2	380	32	14	0,4375
1	2	328	30	-8	-0,266666667
2	2	285	22	5	0,227272727
3	2	368	27	-1	-0,037037037
4	2	332	71	4	0,056338028
5	2	276	22	3	0,136363636
6	2	285	22	3	0,136363636
7	2	325	71	4	0,056338028
8	2	395	67	15	0,223880597
9	2	395	67	24	0,358208955
10	2	389	63	4	0,063492063
11	2	291	38	-3	-0,078947368
12	2	291	38	10	0,263157895
13	2	304	55	3	0,054545455
14	2	303	38	3	0,078947368
15	2	430	61	36	0,590163934
16	2	477	56	5	0,089285714
17	2	424	75	24	0,32
18	2	380	28	7	0,25
1	3	328	79	28	0,35443038
2	3	285	79	19	0,240506329
3	3	368	51	29	0,568627451
4	3	332	60	12	0,2
5	3	276	43	24	0,558139535
6	3	285	78	24	0,307692308
7	3	325	67	12	0,179104478
8	3	395	22	3	0,136363636
9	3	395	74	-1	-0,013513514
10	3	389	80	28	0,35
11	3	291	70	10	0,142857143
12	3	291	44	10	0,227272727
13	3	304	36	-1	-0,027777778
14	3	303	56	12	0,214285714
15	3	430	26	35	1,346153846
16	3	477	78	18	0,230769231
17	3	424	33	1	0,03030303

18	3	380	44	15	0,340909091
1	3	328	34	16	0,470588235
2	3	285	40	13	0,325
3	3	368	79	23	0,291139241
4	3	332	54	16	0,296296296
5	3	276	40	10	0,25
6	3	285	40	10	0,25
7	3	325	54	17	0,314814815
8	3	395	37	15	0,405405405
9	3	395	37	35	0,945945946
10	3	389	45	5	0,111111111
11	3	291	69	9	0,130434783
12	3	291	69	21	0,304347826
13	3	304	69	9	0,130434783
14	3	303	69	9	0,130434783
15	3	430	26	28	1,076923077
16	3	477	56	29	0,517857143
17	3	424	33	17	0,515151515
18	3	380	62	28	0,451612903
1	4	328	48	21	0,4375
2	4	285	24	19	0,791666667
3	4	368	44	19	0,431818182
4	4	332	80	28	0,35
5	4	276	67	26	0,388059701
6	4	285	76	24	0,315789474
7	4	325	45	28	0,622222222
8	4	395	68	35	0,514705882
9	4	395	35	35	1
10	4	389	71	28	0,394366197
11	4	291	45	-2	-0,044444444
12	4	291	45	5	0,111111111
13	4	304	25	9	0,36
14	4	303	71	7	0,098591549
15	4	430	29	35	1,206896552
16	4	477	69	26	0,376811594
17	4	424	46	4	0,086956522
18	4	380	23	14	0,608695652
1	4	328	26	28	1,076923077
2	4	285	49	19	0,387755102
3	4	368	40	19	0,475
4	4	332	53	28	0,528301887
5	4	276	49	24	0,489795918
6	4	285	49	13	0,265306122
7	4	325	53	28	0,528301887
8	4	395	71	35	0,492957746
9	4	395	71	35	0,492957746
10	4	389	34	13	0,382352941
11	4	291	23	9	0,391304348
12	4	291	23	21	0,913043478

13	4	304	44	9	0,204545455
14	4	303	23	10	0,434782609
15	4	430	32	35	1,09375
16	4	477	37	29	0,783783784
17	4	424	47	16	0,340425532
18	4	380	39	14	0,358974359
1	5	328	25	25	1
2	5	285	35	24	0,685714286
3	5	368	54	25	0,462962963
4	5	332	28	28	1
5	5	276	62	26	0,419354839
6	5	285	49	24	0,489795918
7	5	325	27	28	1,037037037
8	5	395	67	35	0,52238806
9	5	395	23	35	1,52173913
10	5	389	62	21	0,338709677
11	5	291	24	10	0,416666667
12	5	291	24	21	0,875
13	5	304	55	20	0,363636364
14	5	303	49	13	0,265306122
15	5	430	43	35	0,813953488
16	5	477	24	29	1,208333333
17	5	424	67	32	0,47761194
18	5	380	53	28	0,528301887
1	5	328	48	28	0,583333333
2	5	285	64	23	0,359375
3	5	368	46	29	0,630434783
4	5	332	56	28	0,5
5	5	276	64	38	0,59375
6	5	285	64	29	0,453125
7	5	325	56	28	0,5
8	5	395	42	35	0,833333333
9	5	395	42	35	0,833333333
10	5	389	37	37	1
11	5	291	64	10	0,15625
12	5	291	64	21	0,328125
13	5	304	34	9	0,264705882
14	5	303	64	22	0,34375
15	5	430	57	46	0,807017544
16	5	477	66	47	0,712121212
17	5	424	77	26	0,337662338
18	5	380	60	28	0,466666667
10	6	389	72	21	0,291666667
11	6	291	72	21	0,291666667
12	6	291	32	21	0,65625
13	6	304	63	21	0,333333333
14	6	303	72	21	0,291666667
16	6	477	35	29	0,828571429
17	6	424	71	32	0,450704225

18	6	380	56	28	0,5
10	6	389	78	37	0,474358974
11	6	291	49	9	0,183673469
12	6	291	49	21	0,428571429
13	6	304	48	14	0,291666667
14	6	303	49	21	0,428571429
16	6	477	75	47	0,626666667
17	6	424	68	26	0,382352941
18	6	380	56	28	0,5

Disposició circular

CIRCULAR					
Simulació	Màquina	Temps cicle original	Temps màquina	g.absolut	g.relatiu
1	1	268	25	25	1
2	1	240	46	15	0,326086957
3	1	304	50	36	0,72
4	1	282	50	27	0,54
5	1	235	55	17	0,309090909
6	1	243	58	13	0,224137931
7	1	275	77	20	0,25974026
8	1	351	20	47	2,35
9	1	327	66	33	0,5
10	1	315	60	23	0,383333333
11	1	257	30	18	0,6
12	1	247	30	17	0,566666667
13	1	252	22	21	0,954545455
14	1	257	31	20	0,64516129
15	1	402	28	61	2,178571429
16	1	452	39	78	2
17	1	398	52	64	1,230769231
18	1	349	33	44	1,333333333
1	1	268	75	32	0,426666667
2	1	240	43	10	0,23255814
3	1	304	72	28	0,388888889
4	1	282	49	9	0,183673469
5	1	235	43	4	0,093023256
6	1	243	43	12	0,279069767
7	1	275	49	12	0,244897959
8	1	351	71	38	0,535211268
9	1	327	71	8	0,112676056
10	1	315	38	14	0,368421053
11	1	257	73	17	0,232876712
12	1	247	73	20	0,273972603
13	1	252	73	4	0,054794521
14	1	257	73	20	0,273972603
15	1	402	28	51	1,821428571
16	1	452	49	59	1,204081633

17	1	398	41	41	1
18	1	349	55	48	0,872727273
1	2	268	34	1	0,029411765
2	2	240	51	10	0,196078431
3	2	304	57	4	0,070175439
4	2	282	80	13	0,1625
5	2	235	57	10	0,175438596
6	2	243	59	10	0,169491525
7	2	275	57	6	0,105263158
8	2	351	49	39	0,795918367
9	2	327	41	15	0,365853659
10	2	315	63	20	0,317460317
11	2	257	48	13	0,270833333
12	2	247	59	8	0,13559322
13	2	252	79	10	0,126582278
14	2	257	29	13	0,448275862
15	2	402	40	63	1,575
16	2	452	65	79	1,215384615
17	2	398	29	53	1,827586207
18	2	349	32	46	1,4375
1	2	268	30	-13	-0,433333333
2	2	240	22	-6	-0,272727273
3	2	304	27	10	0,37037037
4	2	282	71	7	0,098591549
5	2	235	22	-2	-0,090909091
6	2	243	22	-9	-0,409090909
7	2	275	71	-4	-0,056338028
8	2	351	67	45	0,671641791
9	2	327	67	1	0,014925373
10	2	315	63	-13	-0,206349206
11	2	257	38	13	0,342105263
12	2	247	38	5	0,131578947
13	2	252	55	15	0,272727273
14	2	257	38	15	0,394736842
15	2	402	61	48	0,786885246
16	2	452	56	35	0,625
17	2	398	75	45	0,6
18	2	349	28	35	1,25
1	3	268	79	19	0,240506329
2	3	240	79	17	0,215189873
3	3	304	51	16	0,31372549
4	3	282	60	11	0,183333333
5	3	235	43	8	0,186046512
6	3	243	78	13	0,166666667
7	3	275	67	3	0,044776119
8	3	351	22	29	1,318181818
9	3	327	74	19	0,256756757
10	3	315	80	19	0,2375
11	3	257	70	23	0,328571429

12	3	247	44	13	0,295454545
13	3	252	36	8	0,222222222
14	3	257	56	15	0,267857143
15	3	402	26	52	2
16	3	452	78	55	0,705128205
17	3	398	33	51	1,545454545
18	3	349	44	34	0,772727273
1	3	268	34	18	0,529411765
2	3	240	40	10	0,25
3	3	304	79	38	0,481012658
4	3	282	54	11	0,203703704
5	3	235	40	-2	-0,05
6	3	243	40	12	0,3
7	3	275	54	9	0,166666667
8	3	351	37	39	1,054054054
9	3	327	37	19	0,513513514
10	3	315	45	2	0,044444444
11	3	257	69	16	0,231884058
12	3	247	69	12	0,173913043
13	3	252	69	15	0,217391304
14	3	257	69	11	0,15942029
15	3	402	26	56	2,153846154
16	3	452	56	51	0,910714286
17	3	398	33	51	1,545454545
18	3	349	62	47	0,758064516
1	4	268	48	13	0,270833333
2	4	240	24	11	0,458333333
3	4	304	44	22	0,5
4	4	282	80	23	0,2875
5	4	235	67	14	0,208955224
6	4	243	76	14	0,184210526
7	4	275	45	12	0,266666667
8	4	351	68	52	0,764705882
9	4	327	35	18	0,514285714
10	4	315	71	22	0,309859155
11	4	257	45	24	0,533333333
12	4	247	45	15	0,333333333
13	4	252	25	17	0,68
14	4	257	71	20	0,281690141
15	4	402	29	43	1,482758621
16	4	452	69	53	0,768115942
17	4	398	46	37	0,804347826
18	4	349	23	37	1,608695652
1	4	268	26	15	0,576923077
2	4	240	49	27	0,551020408
3	4	304	40	24	0,6
4	4	282	53	21	0,396226415
5	4	235	49	8	0,163265306
6	4	243	49	8	0,163265306

7	4	275	53	17	0,320754717
8	4	351	71	39	0,549295775
9	4	327	71	11	0,154929577
10	4	315	34	-3	-0,088235294
11	4	257	23	14	0,608695652
12	4	247	23	8	0,347826087
13	4	252	44	14	0,318181818
14	4	257	23	5	0,217391304
15	4	402	32	56	1,75
16	4	452	37	66	1,783783784
17	4	398	47	39	0,829787234
18	4	349	39	37	0,948717949
1	5	268	25	2	0,08
2	5	240	35	5	0,142857143
3	5	304	54	31	0,574074074
4	5	282	28	15	0,535714286
5	5	235	62	6	0,096774194
6	5	243	49	8	0,163265306
7	5	275	27	6	0,222222222
8	5	351	67	46	0,686567164
9	5	327	23	19	0,826086957
10	5	315	62	13	0,209677419
11	5	257	24	17	0,708333333
12	5	247	24	12	0,5
13	5	252	55	18	0,327272727
14	5	257	49	23	0,469387755
15	5	402	43	65	1,511627907
16	5	452	24	54	2,25
17	5	398	67	47	0,701492537
18	5	349	53	44	0,830188679
1	5	268	48	3	0,0625
2	5	240	64	24	0,375
3	5	304	46	12	0,260869565
4	5	282	56	9	0,160714286
5	5	235	64	3	0,046875
6	5	243	64	8	0,125
7	5	275	56	11	0,196428571
8	5	351	42	25	0,595238095
9	5	327	42	11	0,261904762
10	5	315	37	4	0,108108108
11	5	257	64	15	0,234375
12	5	247	64	15	0,234375
13	5	252	34	7	0,205882353
14	5	257	64	9	0,140625
15	5	402	57	48	0,842105263
16	5	452	66	51	0,772727273
17	5	398	77	43	0,558441558
18	5	349	60	44	0,733333333
10	6	315	72	6	0,083333333

11	6	257	72	7	0,097222222
12	6	247	32	2	0,0625
13	6	252	63	13	0,206349206
14	6	257	72	7	0,097222222
16	6	452	35	60	1,714285714
17	6	398	71	49	0,690140845
18	6	349	56	40	0,714285714
10	6	315	78	15	0,192307692
11	6	257	49	16	0,326530612
12	6	247	49	11	0,224489796
13	6	252	48	16	0,333333333
14	6	257	49	16	0,326530612
16	6	452	75	70	0,933333333
17	6	398	68	58	0,852941176
18	6	349	56	50	0,892857143

Annex C. Dades exemple impacte ambiental.

Seqüència	0-6-5-7-2-1-4-3	0-6-5-2-7-1-4-3	0-6-5-2-1-7-4-3	0-5-7-2-1-4-6-3	0-5-7-2-1-4-3-6
Moviment 1 (m)	10	10	10	10	10
Moviment 2 (m)	30	30	30	20	20
Moviment 3 (m)	30	30	30	10	10
Moviment 4 (m)	10	30	30	40	40
Moviment 5 (m)	40	20	30	30	30
Moviment 6 (m)	30	50	30	20	20
Moviment 7 (m)	20	20	30	10	30
Moviment 8 (m)	30	30	30	40	20
Distància total (m)	200	220	220	180	180

0-5-2-7-1-4-6-3	0-5-2-7-1-4-3-6	0-5-2-1-7-4-6-3	0-5-2-1-7-4-3-6	0-6-5-7-2-4-1-3	0-6-5-2-7-4-1-3
10	10	10	10	10	10
20	20	20	20	30	30
30	30	30	30	30	30
20	20	30	30	10	30
50	50	30	30	40	20
20	20	30	30	10	30
10	30	10	30	40	40
40	20	40	20	10	10
200	200	200	200	180	200

0-5-7-2-4-6-1-3	0-5-7-2-4-1-6-3	0-5-7-2-4-1-3-6	0-5-2-7-4-6-1-3	0-5-2-7-4-1-6-3	0-5-2-7-4-1-3-6
10	10	10	10	10	10
20	20	20	20	20	20
10	10	10	30	30	30
40	40	40	20	20	20
10	10	10	30	30	30
10	40	40	10	40	40
50	30	10	50	30	10
10	40	20	10	40	20
160	200	160	180	220	180

0-2-1-4-6-3-5-7	òptim	0-7-2-1-4-6-3-5	0-7-2-1-4-3-6-5	0-2-7-1-4-6-3-5	0-2-7-1-4-3-6-5
10	10	10	10	10	10
10	10	30	30	10	10
30	30	40	40	20	20
20	20	30	30	50	50
10	30	20	20	20	20
40	20	10	30	10	30
10	30	40	20	40	20
10	10	10	30	10	30
140	160	190	210	170	190

0-2-1-7-4-6-3-5	0-2-1-7-4-3-6-5-	0-2-4-6-1-3-5-7	0-2-4-1-6-3-5-7	0-2-4-1-3-6-5-7	0-7-2-4-6-1-3-5
10	10	10	10	10	10
10	10	10	10	10	30
30	30	10	10	10	40
30	30	10	40	40	10
30	30	50	30	10	10
10	30	10	40	20	50
40	20	10	10	30	10
10	30	10	10	10	10
170	190	120	160	140	170

0-7-2-4-1-6-3-5	0-7-2-4-1-3-6-5	0-2-7-4-6-1-3-5	0-2-7-4-1-6-3-5	0-2-7-4-1-3-6-5
10	10	10	10	10
30	30	10	10	10
40	40	20	20	20
10	10	30	30	30
40	40	10	40	40
30	10	50	30	10
40	20	10	40	20
10	30	10	10	30
210	190	150	190	170

Annex D. Algorismes complets.

Cos principal de l'algorisme per a disposició lineal de les màquines

```
Public l As Single


Private Sub cmbtempsmaquines_Click()
nmaquines = txtnombremaquines.Text
tbuit = txtbuit.Text
tple = txtple.Text
ReDim tA(1 To nmaquines)
ReDim tB(1 To nmaquines)
ReDim salts(1 To 1)
Form2.Show
End Sub

Private Sub cmdcalcula_Click()

ReDim OPC(1 To 2 * (nmaquines + 1) + 3, 1 To 1)
ReDim V(1 To 2 * (nmaquines + 1))
ReDim PI(1 To 2 * (nmaquines + 1))
ReDim H(1 To 2 * (nmaquines + 1) + 1, 1 To (2 * (nmaquines - 1) + 2 + 1) * (2 * (nmaquines
- 1) + 2 + 2))

OPC(2 * (nmaquines + 1) + 1, 1) = 1
OPC(1, 1) = 0
OPC(2, 1) = 2
OPC(3, 1) = 1
OPC(4, 1) = 3
OPC(2 * (nmaquines + 1) + 3, 1) = tple + tA(1) + 2 * tbuit + tple + tple + tB(1) + 2 * tbuit
OPC(2 * (nmaquines + 1) + 2, 1) = OPC(2 * (nmaquines + 1) + 3, 1) + (nmaquines + 1 -
OPC(2 * (nmaquines + 1) + 1, 1)) * (tple + tbuit)
x = 0

i = 1
While (OPC(2 * (nmaquines + 1) + 1, i) <> nmaquines)

    Call CrearV(OPC, i, V)
    Call CrearPI(V, PI, OPC, i)
    Call Condicions(PI, cond, OPC, i)
    Call CrearTotsH(OPC, i, H, l)
    Call ComprovarCondicions(H, l, cond)
    Call CalcularNousVertex(OPC, H, l)
    OPC(2 * (nmaquines + 1) + 2, i) = 2147483647
```

```
   Call Cotamin(OPC, i)

Wend

q = 1

While q <= 2 * (nmaquines + 1)
   lblseqopt.Caption = lblseqopt.Caption & Str(OPC(q, i))
   q = q + 1
Wend

End Sub
```

Subfuncions de l'algorisme per a disposició lineal de les màquines

```
Public tA() As Long
Public tB() As Long
Public OPC() As Long
Public V() As Integer
Public PI() As Integer
Public cond(1 To 2) As Integer
Public H() As Integer
Public nmaquines As Single
Public saltadestotal As Single
Public jasaltades As Single
Public salts() As Integer
Public i As Single
Public tbuit As Single
Public tple As Single
Public taturat As Single



Public Sub CrearV(OPC, i, V)
j = 0

While j <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 1
   k = 1
   While k <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 2
      If OPC(k, i) = j Then
         V(j + 1) = k - 1
      Else
      End If
      k = k + 1
```

```
    Wend
    j = j + 1

Wend

End Sub

Public Sub CrearVfill(H, j, V, OPC, i)
m = 0

While m <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 3
    k = 1
    While k <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4
        If H(k, j) = m Then
            V(m + 1) = k - 1
        Else
        End If
        k = k + 1
    Wend
    m = m + 1

Wend

End Sub

Public Sub CrearPI(V, PI, OPC, i)
j = 3
While j <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 2

    If V(j) < V(j - 2) Then
        If j Mod 2 = 0 Then
            PI(j) = 2
        Else
            PI(j) = 1
        End If
    Else
        PI(j) = 0
    End If
    j = j + 1
Wend
j = 3
While PI(j) = 0 And j <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 2
    j = j + 1
Wend

If PI(j) = 1 Then
    PI(1) = 0
```

```
   PI(2) = 2
Else
   PI(1) = 1
   PI(2) = 0
End If

End Sub

Public Sub Condicions(PI, cond, OPC, i)
j = 2 * OPC(2 * (nmaquines + 1) + 1, i) + 2
While PI(j) = 0
   j = j - 1
Wend

If PI(j) = 1 Then
   cond(1) = 2
   cond(2) = 3
ElseIf PI(j) = 2 Then
   cond(1) = 1
   cond(2) = 4
End If

End Sub

Public Sub CrearTotsH(OPC, i, H, l)
a = 2 * OPC(2 * (nmaquines + 1) + 1, i) + 2
b = 2 * OPC(2 * (nmaquines + 1) + 1, i) + 3

l = 1
j = 2
k = 3
While j <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 3
   k = j + 1
   While k <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4
      H(j, l) = a
      H(k, l) = b
      lloc = 0
      m = 1
      While m <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4
         If m <> j And m <> k Then
            H(m, l) = OPC(m - lloc, i)
         Else
            lloc = lloc + 1
         End If
         m = m + 1
      Wend
```

```
      k = k + 1
      l = l + 1
   Wend
   j = j + 1
Wend

k = 2
j = 3
While k <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 3
   j = k + 1
   While j <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4
      H(j, l) = a
      H(k, l) = b
      lloc = 0
      m = 1
      While m <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4
         If m <> j And m <> k Then
            H(m, l) = OPC(m - lloc, i)
         Else
            lloc = lloc + 1
         End If
         m = m + 1
      Wend

      j = j + 1
      l = l + 1

   Wend
   k = k + 1
Wend
l = l - 1
End Sub

Public Sub ComprovarCondicions(H, l, cond)
j = 1

While j <= l
   Call CrearVfill(H, j, V, OPC, i)

   If cond(1) = 2 Then
      If (V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 4) < V(2 * OPC(2 * (nmaquines + 1) + 1, i)
+ 1) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 1) < V(2 * OPC(2 * (nmaquines + 1) + 1,
i) + 3) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 3) < V(2 * OPC(2 * (nmaquines + 1) +
1, i) + 2)) Or (V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 1) < V(2 * OPC(2 * (nmaquines + 1)
+ 1, i) + 3) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 3) < V(2 * OPC(2 * (nmaquines +
1) + 1, i) + 2) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 2) < V(2 * OPC(2 * (nmaquines
+ 1) + 1, i) + 4)) Then
```

```
            H(2 * (nmaquines + 1) + 1, j) = 1
        Else
            H(2 * (nmaquines + 1) + 1, j) = 0
        End If
    ElseIf cond(1) = 1 Then
        If (V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 3) < V(2 * OPC(2 * (nmaquines + 1) + 1, i)
+ 2) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 2) < V(2 * OPC(2 * (nmaquines + 1) + 1,
i) + 4) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 4) < V(2 * OPC(2 * (nmaquines + 1) +
1, i) + 1)) Or (V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 2) < V(2 * OPC(2 * (nmaquines + 1)
+ 1, i) + 4) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 4) < V(2 * OPC(2 * (nmaquines +
1) + 1, i) + 1) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 1) < V(2 * OPC(2 * (nmaquines
+ 1) + 1, i) + 3)) Then
            H(2 * (nmaquines + 1) + 1, j) = 1
        Else
            H(2 * (nmaquines + 1) + 1, j) = 0
        End If
    End If
    j = j + 1
Wend

End Sub

Public Sub CalcularNousVertex(OPC, H, l)
m = 1


While m <= l

If H(2 * (nmaquines + 1) + 1, m) = 1 Then
    ReDim Preserve OPC(1 To 2 * (nmaquines + 1) + 3, 1 To UBound(OPC, 2) + 1)
    k = UBound(OPC, 2)
    r = 1

    While r <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4
        OPC(r, k) = H(r, m)
        r = r + 1
    Wend
    OPC(2 * (nmaquines + 1) + 1, k) = OPC(2 * (nmaquines + 1) + 1, i) + 1
    OPC(2 * (nmaquines + 1) + 3, k) = TCicle(H, m)

    r = 1
    jasaltades = 0
    While r <= UBound(salts, 1)
        If salts(r) <= 2 * OPC(2 * (nmaquines + 1) + 1, k) + 1 Then
            jasaltades = jasaltades + 1
        Else
        End If
```

```
      r = r + 1
   Wend

   If 2 * (nmaquines - OPC(2 * (nmaquines + 1) + 1, k)) * (tple + tbuit) - taturat >= 0 And
OPC(2 * (nmaquines + 1) + 1, k) <> nmaquines Then

      OPC(2 * (nmaquines + 1) + 2, k) = OPC(2 * (nmaquines + 1) + 3, k) + 2 * (nmaquines -
OPC(2 * (nmaquines + 1) + 1, k) - (saltadestotal - jasaltades)) * (tple + tbuit) - taturat
   Else
      OPC(2 * (nmaquines + 1) + 2, k) = OPC(2 * (nmaquines + 1) + 3, k)
   End If

Else
End If

m = m + 1

Wend

End Sub

Public Function TCicle(H, m) As Single

Static tarrib() As Single
Static TCicle2, TCicle3, TCicleAux As Single
ReDim tarrib(1 To 2, 1 To 1)

mov = 2
TCicle = tple
tarrib(1, 1) = TCicle
tarrib(2, 1) = 1
TCicle2 = 0
TCicle3 = 1
TCicleAux = 0

While TCicle3 <> TCicle2

   If TCicle2 <> 0 Then
      ReDim Preserve tarrib(1 To 2, 1 To UBound(tarrib, 2) + 1)
      TCicle = TCicle + tple
      tarrib(1, UBound(tarrib, 2)) = TCicle
      tarrib(2, UBound(tarrib, 2)) = 1
      taturat = 0
   Else
   End If

   While mov <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4
```

```
      If H(mov, m) >= 2 Then
         If H(mov, m) Mod 2 = 0 And tA(H(mov, m) \ 2) = 0 Or H(mov, m) Mod 2 <> 0 And
tB(H(mov, m) \ 2) = 0 Then
            mov = mov + 1
         Else
         End If
      Else
      End If

      If mov > 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4 Then

      Else

         k = mov - 1
         movant = 1
         While k >= 2

            j = 1
            While j <= UBound(salts, 1)
               If salts(j) = H(k, m) Then
                  j = UBound(salts, 1) + 2
               Else
                  j = j + 1
               End If
            Wend

            If j = UBound(salts, 1) + 1 Then
               movant = k
               k = 1
            Else
            End If
            k = k - 1
         Wend


         p = 1
         l = UBound(salts, 1)
         While l >= 1
            If salts(l) = H(mov, m) - 2 * p Then
               p = p + 1
            Else
            End If
            l = l - 1
         Wend
```

```
If H(mov, m) > H(movant, m) + 2 And (H(mov, m) - H(movant, m) <> 2) Then
   dist = H(mov, m) \ 2 - (H(movant, m) + 2) \ 2
ElseIf (H(mov, m) - H(movant, m) = 2 * p) Then
   dist = 0
Else
   dist = -(H(mov, m) \ 2 - (H(movant, m) + 2) \ 2)
End If

TCicle = TCicle + dist * tbuit



If (H(mov, m) - H(movant, m) = 2 * j) Then

   If H(mov, m) Mod 2 = 0 Then
      tespera = tA(H(mov, m) \ 2)
      taturat = taturat + tA(H(mov, m) \ 2)
   Else
      tespera = tB(H(mov, m) \ 2)
      taturat = taturat + tB(H(mov, m) \ 2)
   End If

   p = 1
   l = 1

   While l <= UBound(salts, 1)
      If salts(l) = H(mov, m) + 2 * p Then
         p = p + 1
      Else
      End If
      l = l + 1
   Wend

   TCicle = TCicle + tespera + tple + (p - 1) * tbuit

   trobat = 0
   j = 1

   While j <= UBound(tarrib, 2)
      If tarrib(2, j) = H(mov, m) \ 2 Then
         trobat = j
      Else
      End If
      j = j + 1
   Wend
```

```
    If trobat = 0 Then

       trobat = UBound(tarrib, 2) + 1

    Else
    End If

    If trobat = UBound(tarrib, 2) + 1 Then
       ReDim Preserve tarrib(1 To 2, 1 To UBound(tarrib, 2) + 1)
       tarrib(1, trobat) = TCicle
       tarrib(2, trobat) = H(mov, m) \ 2 + p
    Else
       tarrib(1, trobat) = TCicle
       tarrib(2, trobat) = H(mov, m) \ 2 + p
    End If

 Else
    j = 1
    trobat = 0
    While j <= UBound(tarrib, 2)
       If tarrib(2, j) = H(mov, m) \ 2 Then
          trobat = j
       Else
       End If
       j = j + 1
    Wend

    If trobat = 0 Then

       tespera = 0
       trobat = UBound(tarrib, 2) + 1

    Else

       tespera = TCicle - tarrib(1, trobat)

       If H(mov, m) Mod 2 = 0 Then
          If tA(tarrib(2, trobat)) <= tespera Then
             tespera = 0
          Else
             tespera = tA(tarrib(2, trobat)) - tespera
             taturat = taturat + tespera
          End If
       Else
          If tB(tarrib(2, trobat)) <= tespera Then
             tespera = 0
          Else
```

```
                tespera = tB(tarrib(2, trobat)) - tespera
                taturat = taturat + tespera
              End If
            End If
          End If

          p = 1
          l = 1
          While l <= UBound(salts, 1)
            If salts(l) = H(mov, m) + 2 * p Then
              p = p + 1
            Else
            End If
            l = l + 1
          Wend

          TCicle = TCicle + tespera + tple + (p - 1) * tbuit

          If trobat = UBound(tarrib, 2) + 1 Then
            ReDim Preserve tarrib(1 To 2, 1 To UBound(tarrib, 2) + 1)
            tarrib(1, trobat) = TCicle
            tarrib(2, trobat) = H(mov, m) \ 2 + p
          Else
            tarrib(1, trobat) = TCicle
            tarrib(2, trobat) = H(mov, m) \ 2 + p
          End If

        End If
        mov = mov + 1
      End If

    Wend

    TCicle = TCicle + tbuit * tarrib(2, trobat)

    mov = 2
    TCicle3 = TCicle2
    TCicle2 = TCicle - TCicleAux
facin moltes iteracions
    TCicleAux = TCicle

Wend
TCicle = TCicle2
End Function


Public Sub Cotamin(OPC, i)
```

```
r = 1
i = 1
While r <= UBound(OPC, 2)
   If OPC(2 * (nmaquines + 1) + 2, r) <= OPC(2 * (nmaquines + 1) + 2, i) Then
      i = r
   Else
   End If
   r = r + 1
Wend
End Sub
```

Cos principal de l'algorisme per a disposició lineal de les màquines

```
Public l As Single


Private Sub cmbtempsmaquines_Click()
nmaquines = txtnombremaquines.Text
tbuit = txtbuit.Text
tple = txtple.Text
ReDim tA(1 To nmaquines)
ReDim tB(1 To nmaquines)
ReDim salts(1 To 1)
angle = 2 * 3.14159265358979 / (nmaquines + 1)
radi = Sqr(1 / (2 * (1 - Cos(angle))))
Form2.Show
End Sub

Private Sub cmdcalcula_Click()

ReDim OPC(1 To 2 * (nmaquines + 1) + 3, 1 To 1)
ReDim V(1 To 2 * (nmaquines + 1))
ReDim PI(1 To 2 * (nmaquines + 1))
ReDim H(1 To 2 * (nmaquines + 1) + 1, 1 To (2 * (nmaquines - 1) + 2 + 1) * (2 * (nmaquines
- 1) + 2 + 2))

OPC(2 * (nmaquines + 1) + 1, 1) = 1
OPC(1, 1) = 0
OPC(2, 1) = 2
OPC(3, 1) = 1
OPC(4, 1) = 3
OPC(2 * (nmaquines + 1) + 3, 1) = tple + tA(1) + 2 * tbuit + tple + tple + tB(1) + 2 * tbuit
```

```
OPC(2 * (nmaquines + 1) + 2, 1) = OPC(2 * (nmaquines + 1) + 3, 1) + (nmaquines + 1 -
OPC(2 * (nmaquines + 1) + 1, 1)) * (tple + tbuit)
x = 0

i = 1
While (OPC(2 * (nmaquines + 1) + 1, i) <> nmaquines)

   Call CrearV(OPC, i, V)
   Call CrearPI(V, PI, OPC, i)
   Call Condicions(PI, cond, OPC, i)
   Call CrearTotsH(OPC, i, H, l)
   Call ComprovarCondicions(H, l, cond)
   Call CalcularNousVertex(OPC, H, l)
   OPC(2 * (nmaquines + 1) + 2, i) = 2147483647
   Call Cotamin(OPC, i)

Wend

q = 1

While q <= 2 * (nmaquines + 1)
   lblseqopt.Caption = lblseqopt.Caption & Str(OPC(q, i))
   q = q + 1
Wend

End Sub
```

Subfuncions de l'algorisme per a disposició lineal de les màquines

```
Public tA() As Long
Public tB() As Long
Public OPC() As Long
Public V() As Integer
Public PI() As Integer
Public cond(1 To 2) As Integer
Public H() As Integer
Public nmaquines As Single
Public saltadestotal As Single
Public jasaltades As Single
Public salts() As Integer
Public i As Single
Public tbuit As Single
Public tple As Single
```

```
Public taturat As Single
Public angle As Single
Public radi As Single



Public Sub CrearV(OPC, i, V)
j = 0

While j <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 1
   k = 1
   While k <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 2
      If OPC(k, i) = j Then
         V(j + 1) = k - 1
      Else
      End If
      k = k + 1
   Wend
   j = j + 1

Wend

End Sub

Public Sub CrearVfill(H, j, V, OPC, i)
m = 0

While m <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 3
   k = 1
   While k <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4
      If H(k, j) = m Then
         V(m + 1) = k - 1
      Else
      End If
      k = k + 1
   Wend
   m = m + 1

Wend

End Sub

Public Sub CrearPI(V, PI, OPC, i)
j = 3
While j <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 2
```

```
    If V(j) < V(j - 2) Then
        If j Mod 2 = 0 Then
            PI(j) = 2
        Else
            PI(j) = 1
        End If
    Else
        PI(j) = 0
    End If
    j = j + 1
Wend
j = 3
While PI(j) = 0 And j <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 2
    j = j + 1
Wend

If PI(j) = 1 Then
    PI(1) = 0
    PI(2) = 2
Else
    PI(1) = 1
    PI(2) = 0
End If

End Sub

Public Sub Condicions(PI, cond, OPC, i)
j = 2 * OPC(2 * (nmaquines + 1) + 1, i) + 2
While PI(j) = 0
    j = j - 1
Wend

If PI(j) = 1 Then
    cond(1) = 2
    cond(2) = 3
ElseIf PI(j) = 2 Then
    cond(1) = 1
    cond(2) = 4
End If

End Sub

Public Sub CrearTotsH(OPC, i, H, l)
A = 2 * OPC(2 * (nmaquines + 1) + 1, i) + 2
b = 2 * OPC(2 * (nmaquines + 1) + 1, i) + 3
```

```
l = 1
j = 2
k = 3
While j <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 3
   k = j + 1
   While k <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4
      H(j, l) = A
      H(k, l) = b
      lloc = 0
      m = 1
      While m <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4
         If m <> j And m <> k Then
            H(m, l) = OPC(m - lloc, i)
         Else
            lloc = lloc + 1
         End If
         m = m + 1
      Wend

      k = k + 1
      l = l + 1
   Wend
   j = j + 1
Wend

k = 2
j = 3
While k <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 3
   j = k + 1
   While j <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4
      H(j, l) = A
      H(k, l) = b
      lloc = 0
      m = 1
      While m <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4
         If m <> j And m <> k Then
            H(m, l) = OPC(m - lloc, i)
         Else
            lloc = lloc + 1
         End If
         m = m + 1
      Wend

      j = j + 1
      l = l + 1

   Wend
```

```
   k = k + 1
Wend
l = l - 1
End Sub

Public Sub ComprovarCondicions(H, l, cond)
j = 1

While j <= l
   Call CrearVfill(H, j, V, OPC, i)
   If cond(1) = 2 Then
      If (V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 4) < V(2 * OPC(2 * (nmaquines + 1) + 1, i)
+ 1) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 1) < V(2 * OPC(2 * (nmaquines + 1) + 1,
i) + 3) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 3) < V(2 * OPC(2 * (nmaquines + 1) +
1, i) + 2)) Or (V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 1) < V(2 * OPC(2 * (nmaquines + 1)
+ 1, i) + 3) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 3) < V(2 * OPC(2 * (nmaquines +
1) + 1, i) + 2) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 2) < V(2 * OPC(2 * (nmaquines
+ 1) + 1, i) + 4)) Then
         H(2 * (nmaquines + 1) + 1, j) = 1
      Else
         H(2 * (nmaquines + 1) + 1, j) = 0
      End If
   ElseIf cond(1) = 1 Then
      If (V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 3) < V(2 * OPC(2 * (nmaquines + 1) + 1, i)
+ 2) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 2) < V(2 * OPC(2 * (nmaquines + 1) + 1,
i) + 4) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 4) < V(2 * OPC(2 * (nmaquines + 1) +
1, i) + 1)) Or (V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 2) < V(2 * OPC(2 * (nmaquines + 1)
+ 1, i) + 4) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 4) < V(2 * OPC(2 * (nmaquines +
1) + 1, i) + 1) And V(2 * OPC(2 * (nmaquines + 1) + 1, i) + 1) < V(2 * OPC(2 * (nmaquines
+ 1) + 1, i) + 3)) Then
         H(2 * (nmaquines + 1) + 1, j) = 1
      Else
         H(2 * (nmaquines + 1) + 1, j) = 0
      End If
   End If
   j = j + 1
Wend

End Sub

Public Sub CalcularNousVertex(OPC, H, l)
m = 1


While m <= l

If H(2 * (nmaquines + 1) + 1, m) = 1 Then
```

```
   ReDim Preserve OPC(1 To 2 * (nmaquines + 1) + 3, 1 To UBound(OPC, 2) + 1)
   k = UBound(OPC, 2)
   r = 1

   While r <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4
      OPC(r, k) = H(r, m)
      r = r + 1
   Wend
   OPC(2 * (nmaquines + 1) + 1, k) = OPC(2 * (nmaquines + 1) + 1, i) + 1
   OPC(2 * (nmaquines + 1) + 3, k) = TCicle(H, m)

   r = 1
   jasaltades = 0
   While r <= UBound(salts, 1)
      If salts(r) <= 2 * OPC(2 * (nmaquines + 1) + 1, k) + 1 Then
         jasaltades = jasaltades + 1
      Else
      End If
      r = r + 1
   Wend

   If 2 * (nmaquines - OPC(2 * (nmaquines + 1) + 1, k)) * (tple + tbuit) - taturat >= 0 And
OPC(2 * (nmaquines + 1) + 1, k) <> nmaquines Then
      OPC(2 * (nmaquines + 1) + 2, k) = OPC(2 * (nmaquines + 1) + 3, k) + 2 * (nmaquines -
OPC(2 * (nmaquines + 1) + 1, k) - (saltadestotal - jasaltades)) * (tple + tbuit) - taturat
   Else
      OPC(2 * (nmaquines + 1) + 2, k) = OPC(2 * (nmaquines + 1) + 3, k)
   End If

Else
End If

m = m + 1

Wend

End Sub

Public Function TCicle(H, m) As Single

Static tarrib() As Single
Static TCicle2, TCicle3, TCicleAux As Single
ReDim tarrib(1 To 2, 1 To 1)

mov = 2
TCicle = tple
tarrib(1, 1) = TCicle
```

```
tarrib(2, 1) = 1
TCicle2 = 0
TCicle3 = 1
TCicleAux = 0

While TCicle3 <> TCicle2

   If TCicle2 <> 0 Then
      ReDim Preserve tarrib(1 To 2, 1 To UBound(tarrib, 2) + 1)
      TCicle = TCicle + tple
      tarrib(1, UBound(tarrib, 2)) = TCicle
      tarrib(2, UBound(tarrib, 2)) = 1
      taturat = 0
   Else
   End If

   While mov <= 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4

      If H(mov, m) >= 2 Then
         If H(mov, m) Mod 2 = 0 And tA(H(mov, m) \ 2) = 0 Or H(mov, m) Mod 2 <> 0 And
tB(H(mov, m) \ 2) = 0 Then
            mov = mov + 1
         Else
         End If
      Else
      End If

      If mov > 2 * OPC(2 * (nmaquines + 1) + 1, i) + 4 Then

      Else

         k = mov - 1
         movant = 1
         While k >= 2
            j = 1
            While j <= UBound(salts, 1)
               If salts(j) = H(k, m) Then
                  j = UBound(salts, 1) + 2
               Else
                  j = j + 1
               End If
            Wend

            If j = UBound(salts, 1) + 1 Then
               movant = k
               k = 1
            Else
```

```
    End If
    k = k - 1
Wend


p = 1
l = UBound(salts, 1)
While l >= 1
    If salts(l) = H(mov, m) - 2 * p Then
        p = p + 1
    Else
    End If
    l = l - 1
Wend



If H(mov, 1) - 2 * p = H(movant, 1) Then
    dist = 0
Else
    A = angle * Abs((H(mov, 1) \ 2 - (H(movant, 1) \ 2 + 1)))
    dist = Abs(Sqr(2 * radi * radi * (1 - Cos(A))))
End If

TCicle = TCicle + dist * tbuit


If (H(mov, m) - H(movant, m) = 2 * j) Then

    If H(mov, m) Mod 2 = 0 Then
        tespera = tA(H(mov, m) \ 2)
        taturat = taturat + tA(H(mov, m) \ 2)
    Else
        tespera = tB(H(mov, m) \ 2)
        taturat = taturat + tB(H(mov, m) \ 2)
    End If

    p = 1
    l = 1

    While l <= UBound(salts, 1)
        If salts(l) = H(mov, m) + 2 * p Then
            p = p + 1
        Else
        End If
        l = l + 1
    Wend
```

```
            A = angle * (1 + (p - 1))
            dist = Abs(Sqr(2 * radi * radi * (1 - Cos(A))))

            TCicle = TCicle + tespera + tple + (dist - Abs(Sqr(2 * radi * radi * (1 -
Cos(angle))))) * tbuit
            trobat = 0
            j = 1

            While j <= UBound(tarrib, 2)
               If tarrib(2, j) = H(mov, m) \ 2 Then
                  trobat = j
               Else
               End If
               j = j + 1
            Wend

            If trobat = 0 Then

               trobat = UBound(tarrib, 2) + 1

            Else
            End If

            If trobat = UBound(tarrib, 2) + 1 Then
               ReDim Preserve tarrib(1 To 2, 1 To UBound(tarrib, 2) + 1)
               tarrib(1, trobat) = TCicle
               tarrib(2, trobat) = H(mov, m) \ 2 + p
            Else
               tarrib(1, trobat) = TCicle
               tarrib(2, trobat) = H(mov, m) \ 2 + p
            End If

         Else
            j = 1
            trobat = 0
            While j <= UBound(tarrib, 2)
               If tarrib(2, j) = H(mov, m) \ 2 Then
                  trobat = j
               Else
               End If
               j = j + 1
            Wend

            If trobat = 0 Then

               tespera = 0
```

```
            trobat = UBound(tarrib, 2) + 1

        Else

            tespera = TCicle - tarrib(1, trobat)

            If H(mov, m) Mod 2 = 0 Then
                If tA(tarrib(2, trobat)) <= tespera Then
                    tespera = 0
                Else
                    tespera = tA(tarrib(2, trobat)) - tespera
                    taturat = taturat + tespera
                End If
            Else
                If tB(tarrib(2, trobat)) <= tespera Then
                    tespera = 0
                Else
                    tespera = tB(tarrib(2, trobat)) - tespera
                    taturat = taturat + tespera
                End If
            End If
        End If

        p = 1
        l = 1
        While l <= UBound(salts, 1)
            If salts(l) = H(mov, m) + 2 * p Then
                p = p + 1
            Else
            End If
            l = l + 1
        Wend

         A = angle * (1 + (p - 1))
        dist = Abs(Sqr(2 * radi * radi * (1 - Cos(A))))

        TCicle = TCicle + tespera + tple + (dist - Abs(Sqr(2 * radi * radi * (1 -
Cos(angle))))) * tbuit

        If trobat = UBound(tarrib, 2) + 1 Then
            ReDim Preserve tarrib(1 To 2, 1 To UBound(tarrib, 2) + 1)
            tarrib(1, trobat) = TCicle
            tarrib(2, trobat) = H(mov, m) \ 2 + p
        Else
            tarrib(1, trobat) = TCicle
            tarrib(2, trobat) = H(mov, m) \ 2 + p
        End If
```

```
            End If
            mov = mov + 1
        End If

    Wend
    A = angle * tarrib(2, trobat)
    dist = Abs(Sqr(2 * radi * radi * (1 - Cos(A))))

    TCicle = TCicle + tbuit * dist
    mov = 2
    TCicle3 = TCicle2
    TCicle2 = TCicle - TCicleAux
    TCicleAux = TCicle

Wend
TCicle = TCicle2
End Function


Public Sub Cotamin(OPC, i)
r = 1
i = 1
While r <= UBound(OPC, 2)
    If OPC(2 * (nmaquines + 1) + 2, r) <= OPC(2 * (nmaquines + 1) + 2, i) Then
        i = r
    Else
    End If
    r = r + 1
Wend
End Sub
```